基于 MILP 方法的 LED 密码安全性分析 *

刘波涛 [a, b]，彭长根 [a, b]，吴睿雪 [a, c]，丁红发 [b, d]，谢明明 [c, d]

(贵州大学 a. 计算机科学与技术学院, b. 贵州省公共大数据重点实验室, c. 密码学与数据安全研究所, d. 数学与统计学院, 贵阳 550025)

摘　要：基于自动化搜索算法求解差分特征与线性逼近，成为了分组密码的差分与线性攻击研究热点，提出一种面向半个字节 MILP 模型自动化搜索密码算法的差分特征与线性逼近方法，进行对轻量级 LED 密码实现分析，以较少的变量与约束不等式求解活跃 S 盒数量，4 轮运算至少有 25 个活跃 S 盒，这个结果与算法设计者给出的活跃 S 盒理论值相同，验证了该方法的正确性。最后，计算 LED 算法的最大差分特征及线性逼近概率，证明其能够抵抗差分与线性攻击。

关键词：分组密码；差分攻击；线性攻击；MILP 模型；LED 密码

中图分类号：TP309.2　　doi: 10.19734/j.issn.1001-3695.2018.07.0563

Based on MILP method for security analysis of LED

Liu Botao[a, b], Peng Changgen[a, b], Wu Ruixue[a, c], Ding Hongfa[b, d], Xie Mingming[c, d]

(a. *College of Computer Science & Technology*, b. *Guizhou Province Key Laboratory of Public Big Data*, c. *Institute of Cryptography & Data Security*, d. *College of Mathematics & Statistics*, *Guizhou University*, *Guiyang* 550025, *China*)

Abstract: Based on the automatic search algorithm to solve differential characteristic and linear characteristic, it has become hot topics in differential attack and linear attack. This paper presented a differential characteristic and linear characteristic method based on the half-byte MILP model for automatic search cryptanalysis. This method is used to analyze for the LED lightweight block cipher, to solve the number of active S-boxes with fewer variables and constraint inequalities. The 4 rounds of LED obtained at least 25 active S-boxes, which is same as theoretical value given by the LED designer, verifying the correctness of the half-byte MILP model. Finally, this paper calculated the maximum differential characteristic probability and the maximum linear characteristic probability of LED, which proves that it can resist differential attack and linear attack.

Key words: block cipher; differential attack; Linear attack; MILP; LED

0　引言

近年来，轻量级分组密码作为物联网设备信息安全保障，其设计与分析引起了学术与工业界的高度关注。目前，在国内外已提出了一些轻量级分组密码算法，其中知名的密码算法有 PRESENT[1]、LED[2]、LBlock[3]、KLEIN[4]、RECTANGLE[5]、SKINNY[6]、SFN[7]等。轻量级 LED 分组密码是由南洋理工大学郭建等人在 2011 年的 CHES 上被提出，算法设计兼顾了软硬件实现性能的目标，软件运行效率快，在硬件方面，设计了超轻量级（甚至于无）密钥编排，密钥直接通过单输入触发器实现，节省大量硬件面积资源，这种超轻量级密钥编排方式得到了 SKINNY 等多个算法学习及采用，与 PRESENT、LBlock、KLEIN、RECTANGLE 密码算法相比，其面积更小。为了实现微型无线传感器等设备中无线传感器网络（wireless sensor networks, WSN）[8]与车载 Ad-Hoc 网络（vehicular ad-hoc networks, VANET）[9]传输信息的机密性、完整性和认证性，以及保护移动智能设备中用户隐私[10]，作为轻量级分组密码算法中的佼佼者 LED 算法具有适应能力强、软硬件执行效率高等优点，非常适合应用于这些物联网中智能微型设备。

轻量级分组密码算法是硬件实现具有资源少、功耗低等优点，但其安全性一直是密码学者关注的焦点。对于分组密码而言，差分攻击[11]与线性攻击[12]是两种强有力的攻击方法，从而抵抗差分与线性攻击是评估算法设计安全性一个非常重要的指标，所以在密码算法设计的安全分析评估中，设计者首要评估差分攻击与线性攻击，来验证密码算法的安全性。在差分攻击与线性攻击中，关键是寻找高概率的差分特性与线性逼近；对于最大差分特性概率与最大线性逼近概率是通过分析算法存在最小活跃 S 盒的数量来进行计算。目前，自动化搜索算法来获取活跃 S 盒最小值，成为差分攻击与线性攻击的研究热点。2011 年，Mouha 等人[13]将混合整数线性规划（mixed integer linear programming , MILP）问题应用于自动化求解最小活跃 S 盒，该方法以字节作为差分变量单位，结合 SPN（Substitution-Permutation Network）结构与 Feistel

---

收稿日期：2018-07-25；**修回日期：**2018-09-11　　**基金项目：**国家自然科学基金资助项目（61662009，61772008）；国家“十三五”密码发展基金资助项目（MMJJ20170129）；贵州省科技计划资助项目（黔科合基础［2016］2315，黔科合基础［2017］1045，黔科合重大专项字［2017］3002，黔科合重大专项字［2018］3001）；湖南省自然科学基金资助项目（2017JJ2010）

作者简介：刘波涛（1991-），男，湖南株洲人，硕士研究生，主要研究方向为分组密码与大数据安全（teslal0505@foxmail.com）；彭长根（1963-），男（侗族），贵州贵阳人，教授，博导，博士，主要研究方向为隐私保护、密码学与大数据安全；吴睿雪（1995-），女，四川西昌人，硕士研究生，主要研究方向为隐私保护与大数据安全；丁红发（1988-），男，贵州贵阳人，讲师，博士研究生，主要研究方向为分组密码、访问控制及大数据安全；谢明明（1993-），男，湖北荆州人，硕士研究生，主要研究方向为隐私保护与大数据安全.

结构算法的轮函数，对字节级混淆与扩散模块组件进行约束分析、编程实现求解。2014 年，Sun 等人[14]在 ASIACRYPT 上，对 MILP 自动化求解方法进行了扩展，将原有字节级的差分变量单位扩展到了比特级的差分变量单位。2016 年，Fu 等人[15]在 FSE 上，提出了关于 ARX 结构分组密码算法的 MILP 自动化求解方法，分析了轻量级分组 SIMON 算法[16]的差分特性与线性逼近。2017 年，尹军等人[17,18]在文献[14]方法基础上，自动化搜索了轻量级分组 ESF 算法[19]的差分特征与线性逼近，并给出了全轮 ESF 能够抵抗差分攻击。目前，对于 LED 密码算法的安全性分析主要基于传统理论方法求解差分特征与线性逼近[20]，并且 LED 密码算法的设计者没有给出具体的推断与计算过程去证明。在 MILP 自动化搜索模型求解 LED 算法的差分特征与线性逼近方面，国内外尚未发现有公开发表的成果。因此，对于保护物联网设备敏感信息的安全性，利用 MILP 模型自动化搜索来进行基于半个字节的 LED 算法及其他密码安全性分析，是加强自主密码算法设计与分析提供了非常重要的价值。

本文提出一种面向半个字节 MILP 模型自动化搜索密码算法的差分特征与线性逼近方法，该方法通过编程实现，该模型以较少的变量与约束不等式求解算法的活跃 S 盒数量，提升了 MILP 自动化分析技术的效率，精确地验证了 LED 算法活跃 S 盒的数量以及最大差分特征概率与最大线性逼近概率，证明了 LED 密码算法是能够抵抗差分攻击与线性攻击。

1 LED 算法描述

1.1 常用的符号与术语

P :64 位明文

C :64 位密文

K :原始密钥

SK :64 位轮密钥

$\oplus$:异或运算

$\|$: 链接符

$state$:中间状态值

rc :轮常数值

1.2 LED 算法

LED 是一种典型的 SPN 结构轻量级分组密码算法，其分组为 64 比特，密钥分别为 64 比特与 128 比特两种，对应记为 LED-64 与 LED-128，迭代轮数 r 分别为 32 轮和 48 轮。

算法加密过程中，轮函数包含 5 个模块组件：轮密钥加变换（AddRoundkey）、常数加变换（AddConstants）、S 盒替换变换（SubCells）、行移位变换（ShiftRows）及列混合变换（MixColumnsSerial），LED 算法加密流程如图 1 所示。LED 算法是以半个字节（$\mathbb{Z}_2^4$）为运算操作单位，明文输入符号为 $P(p_0,p_1,\cdots,p_{15})$，密文输出符号为 $C(c_0,c_1,\cdots,c_{15})$。

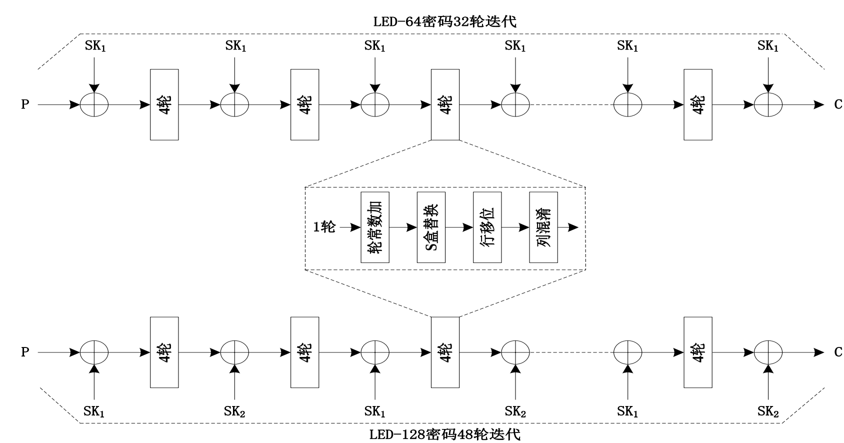


图 1 LED 算法加密流程

Fig. 1 The encryption process of LED

下面对 LED 算法的轮函数 5 个模块组件及轮密钥生成方案进行描述与分析：

1）轮密钥加变换（AddRoundkey）

轮密钥加变换是指中间状态值（$state$）与轮密钥（SK）进行比特间异或操作，LED 算法是每 4 轮进行一次轮密钥加变换操作。LED-64 算法与 LED-128 算法轮函数主要差异是轮密钥加变换操作不同，具体如下所示。

LED-64 算法的轮密钥加变换如下公式所示：

$$state_{[0:63]} \leftarrow state_{[0:63]} \oplus SK_{1[0:63]} \tag{1}$$

LED-128 算法中，奇数次 4 轮运算轮密钥加操作如下所示：

$$state_{[0:63]} \leftarrow state_{[0:63]} \oplus SK_{1[0:63]} \tag{2}$$

偶数次 4 轮运算轮密钥加操作如下所示：

$$state_{[0:63]} \leftarrow state_{[0:63]} \oplus SK_{2[0:63]} \tag{3}$$

2）常数加变换（AddConstants）

常数加变换是 32 比特的中间状态值与固定常数（0、1、2、3 及轮常数 rc）进行异或运算，该轮常数 rc 见文献[2]，运算过程如下所示。

$$state_{i[0:7]} \leftarrow state_{i[0:3]} \oplus j \,\|\, state_{i[4:7]} \oplus rc \tag{4}$$

其中：$(i=0,1,4,5,8,9,12,13)$ 及 $(j=0,1,2,3)$。

3）S 盒替换变换（SubCells）

S 盒替换变换是 LED 算法的一个非线性层变换，LED 算法的 S 盒采用于 PRESENT 算法 4×4 加密 S 盒，S 盒见表 1，S 盒替换操作过程如下所示：

$$state_{i[0:3]} \leftarrow S-box\left(state_{i[0:3]}\right) \quad (0 \le i \le 15) \tag{5}$$

表 1 S 盒元素（16 进制）

Table 1 S-box in hexadecimal form

x	0	1	2	3	4	5	6	7
S-box(x)	C	5	6	B	9	0	A	D
x	8	9	A	B	C	D	E	F
S-box(x)	3	E	F	8	4	7	1	2

4）行移位变换（ShiftRows）

行移位变换是进行半字节块的循环移位操作，在 64 bit

构成的 4×4 半字节方正中，第 1 行半字节块不移动，第 2 行循环左移 1 个半字节块，第 3 行循环左移 2 个半字节块，第 4 行循环左移 3 个半字节块，如图 2 所示。

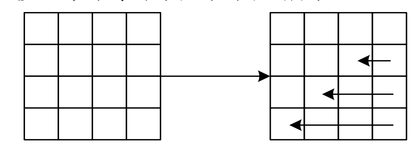


图 2　行移位变换

Fig. 2　Operation of shiftrows

5）列混合变换（MixColumnsSerial）

列混合变换是基于有限域 $GF(2^4)$ 的 4×4 半字节中间状态值矩阵与混合固定矩阵进行相乘运算，具体过程如下所示：

$$state_{[0:63]}\leftarrow\begin{bmatrix}4&1&2&2\\8&6&5&6\\B&E&A&9\\2&2&F&B\end{bmatrix}\bullet\begin{bmatrix}state_{0[0:3]}&state_{1[0:3]}&state_{2[0:3]}&state_{3[0:3]}\\state_{4[0:3]}&state_{5[0:3]}&state_{6[0:3]}&state_{7[0:3]}\\state_{8[0:3]}&state_{9[0:3]}&state_{10[0:3]}&state_{11[0:3]}\\state_{12[0:3]}&state_{13[0:3]}&state_{14[0:3]}&state_{15[0:3]}\end{bmatrix}\tag{6}$$

6）轮密钥生成方案

LED 算法的轮密钥生成方案是一个不需要进行扩展运算的，在 LED-64 算法中，是直接将 64 位原始密钥赋值给轮密钥 SK_1，表示为 $SK_1=K$；在 LED-128 算法中，是先将 128 位原始密钥均等分为两个部分，在分别赋值给轮密钥 SK_1 与 SK_2，$SK_1\|SK_2=K$。

2　面向半个字节的 MILP 模型

MILP 问题是运筹学中的一类优化问题，目的是在线性约束条件下求解目标函数的最大值或者最小值。在构造自动化分析程序的差分特征中，每一个半字节的差分值分为 $\Delta=0$ 与 $\Delta\neq0$ 两种情况，利用变量 0 和 1 进行描述 r 轮的密码算法差分特征，并且差分变量 x 来描述 r 轮的每一个半字节差分值状态。

定义 1　在字符串中，n 个半字节差分表示为 $\Delta=(\Delta_0,\Delta_1,\dots,\Delta_{n-1})$，然后半字节差分变量 $x=(x_0,x_1,\dots,x_{n-1})$，当半字节差分值 $\Delta_i=0$，则差分变量取值 $x_i=0$；当差分值 $\Delta_i\neq0$，则差分变量取值 $x_i=1$，其中 $(0\le i\le n-1)$。

分组密码中，算法轮函数是具有较好的混淆与扩散效果，一般非线性组件是使算法达到混淆作用，线性组件是使算法达到扩散效果。

a）非线性变换的约束。非线性变换是采用于 S 盒替换操作，而在轻量级分组密码算法中，为了降低实现成本，采用 4×4 的 S 盒，S 盒替换变换就是进行半个字节的操作。假设 S 盒的 4 比特（半个字节）输入差分表示为 $(\Delta a_0,\Delta a_1,\Delta a_2,\Delta a_3)$，则 S 盒的 4 比特（半个字节）输出差分表示为 $(\Delta b_0,\Delta b_1,\Delta b_2,\Delta b_3)$，用 A_t 表示这个 4×4 的 S 盒的活跃状态。

在非线性变换的约束中，r 轮差分特征中至少有一个 S 盒是活跃 S 盒，保证 Δa_0，Δa_1，Δa_2，Δa_3 有任意一个状态为 1，则 $A_t=1$，用不等式约束为

$$\begin{cases}\Delta a_0-A_t\le0\\\Delta a_1-A_t\le0\\\Delta a_2-A_t\le0\\\Delta a_3-A_t\le0\end{cases}\tag{7}$$

反之，当 $A_t=1$ 时，则 Δa_0，Δa_1，Δa_2，Δa_3 必定存在一个非零，用不等式约束为

$$\Delta a_0+\Delta a_1+\Delta a_2+\Delta a_3-A_t\ge0\tag{8}$$

分析中，对于一个双射 4×4 的 S 盒，当输入半字节差分是非零的，输出半字节差分必然也是非零的，输入半字节差分是零，输出半字节差分必然也是零，用不等式约束为

$$\begin{cases}4\sum_{j=0}^{3}\Delta b_j-\sum_{j=0}^{3}\Delta a_j\ge0\\4\sum_{j=0}^{3}\Delta a_j-\sum_{j=0}^{3}\Delta b_j\ge0\end{cases}\tag{9}$$

根据算法运算分组单位与结构不同，这些变量的约束也是不相同的。

b）线性变换的约束：线性变换 L，在线性变换 L 当中，给定差分分支数为 B。假设线性变换输入半字节差分变量表示为 $(x^L_{in_0},x^L_{in_1},\cdots,x^L_{in_{n-1}})$，则输出半字节差分变量表示为 $(x^L_{out_0},x^L_{out_1},\cdots,x^L_{out_{n-1}})$，假设一个 d^L 是 0 与 1 的变量，当线性变换中，$x^L_{in_0},x^L_{in_1},\cdots,x^L_{in_{n-1}},x^L_{out_0},x^L_{out_1},\cdots,x^L_{out_{n-1}}$ 都为 0 时，则 d^L 取值为 0，否则，d^L 取值为 1。线性变换用不等式约束表示为

$$\begin{aligned}&x^L_{in_0}+x^L_{in_1}+\cdots+x^L_{in_{n-1}}+x^L_{out_0}+x^L_{out_1}+\cdots+x^L_{out_{n-1}}\ge Bd^L\\&d^L\ge x^L_{in_0},\\&\cdots\\&d^L\ge x^L_{in_{n-1}},\\&d^L\ge x^L_{out_0},\\&\cdots\\&d^L\ge x^L_{out_{n-1}}\end{aligned}\tag{10}$$

3　建立 LED 算法的自动化 MILP 模型

在 LED 密码中，轮函数包括这 5 个模块的变换：轮密钥加、常数加、S 盒替换、行移位及列混合，这些模块的运算是基于半个字节分组变换运算。这 5 个模块中，具有混淆作用的模块是 S 盒替换变换，具有线性扩散作用的模块是行移位变换与列混合变换。针对 LED 算法是基于半个字节分组变换运算，建立面向半个字节的自动化 MILP 模型。在 LED 算法中，根据算法结构特点，建立自动化 MILP 模型，重点考虑是如下几个操作：

a）S 盒替换变换：$\mathbb{F}_2^4\to\mathbb{F}_2^4$。

b）行移位变换与列混淆变换：$\mathbb{F}_2^{64}\to\mathbb{F}_2^{64}$。

S 盒替换变换的 MILP 模型约束，由于 LED 算法的密码生成方案不进行运算操作，从而主要是分析算法加密过程使用的 S 盒替换，结合第 2 节中的描述，一个双射的 4×4 S 盒差分分析中，半字节的输入差分是非零，则半字节的输出差分也是非零，此时差分活跃 S 盒表示为 1，反之亦然；用不等式约束为

$$At=\begin{cases}1&\Delta x\neq0\\0&\Delta x=0\end{cases}\tag{11}$$

对于行移位变换与列混合变换线性扩散层的 MILP 模型约束，在列混合变换中，一个 n 阶的 MDS 型矩阵，这个 n 阶 MDS 型矩阵的最大分支数是 $n+1$（分支数从理论上可以给出差分攻击与线性攻击的抵抗界限，扩散层分支数越大，扩散效果越好），LED 算法的固定混合矩阵是一个 4 阶 MDS 型的矩阵，线性扩散层达到最佳，从而 LED 算法差分与线性的分支数 $B=5$，设 d 为一个 0 与 1 的变量。

行移位变换的 MILP 模型约束，在行移位变换当中，只是进行半个字节的循环移位操作，没有改变差分变量值，不

产生新的差分变量，变化情况如下所示：

$$\begin{bmatrix} x_0 & x_4 & x_8 & x_{12} \\ x_1 & x_5 & x_9 & x_{13} \\ x_2 & x_6 & x_{10} & x_{14} \\ x_3 & x_7 & x_{11} & x_{15} \end{bmatrix} \xrightarrow{SR} \begin{bmatrix} x_0 & x_4 & x_8 & x_{12} \\ x_5 & x_9 & x_{13} & x_1 \\ x_{10} & x_{14} & x_2 & x_6 \\ x_{15} & x_3 & x_7 & x_{11} \end{bmatrix} \quad (12)$$

行移位变换的 MILP 模型约束 C 语言实现代码描述如下：

```
{
  int state[4];
  for(j = 1; j < 4; i++)
  {
    for(i = 0; i <4; i++)
      state[i] = a[j][(i + j) % 4];
    for(i = 0; i <4; i++)
      a[j][i] = state[i];
  }
}
```

列混合变换的 MILP 模型约束，在列混合变换是有限域 $GF(2^4)$ 上的矩阵相乘操作，矩阵单元中每一列的每个元素值发生了变化，差分变量值也得到了改变，产生新的差分变量，改变情况如下所示：

$$\begin{bmatrix} x_0 & x_4 & x_8 & x_{12} \\ x_5 & x_9 & x_{13} & x_1 \\ x_{10} & x_{14} & x_2 & x_6 \\ x_{15} & x_3 & x_7 & x_{11} \end{bmatrix} \xrightarrow{MC} \begin{bmatrix} x_{16} & x_{20} & x_{24} & x_{28} \\ x_{17} & x_{21} & x_{25} & x_{29} \\ x_{18} & x_{22} & x_{26} & x_{30} \\ x_{19} & x_{23} & x_{27} & x_{31} \end{bmatrix} \quad (13)$$

列混合变换的 MILP 模型约束 C 语言实现代码描述如下：

```
void MixColumnsSerial (int a[4][4])
{
  for(i=0; i <4 ; i++)
  {
    for(j = 0; j < 4; j++)
      printf("x%j +", a[j][i]);
    for(j = 0; j < 3; j++)
      printf("x%j +", next + j);
      printf("x%j - 5 d%j >= 0\n", next+3,dummy);
    for(j = 0; j < 4; j++)
      printf("d%j - x%j >= 0\n",
      dummy, a[j][i]);
    for(j = 0; j < 4; j++)
      printf("d%j - x%j >= 0\n",
      dummy, a[j][i] = next++);
    dummy++;
  }
}
```

LED 算法是每 4 轮作为一个大的运算步骤，在这 4 轮运算中，充分使得算法得到混淆与扩散，本文以 LED 算法的一个 4 轮运算进行具体自动化 MILP 模型约束分析，算法 4 轮变换的混淆与扩散情况，如式（14）所示。

$$\begin{aligned}
&\begin{bmatrix} x_0 & x_4 & x_8 & x_{12} \\ x_1 & x_5 & x_9 & x_{13} \\ x_2 & x_6 & x_{10} & x_{14} \\ x_3 & x_7 & x_{11} & x_{15} \end{bmatrix} \xrightarrow{SC} \begin{bmatrix} x_0 & x_4 & x_8 & x_{12} \\ x_1 & x_5 & x_9 & x_{13} \\ x_2 & x_6 & x_{10} & x_{14} \\ x_3 & x_7 & x_{11} & x_{15} \end{bmatrix} \xrightarrow{SR} \begin{bmatrix} x_0 & x_4 & x_8 & x_{12} \\ x_5 & x_9 & x_{13} & x_1 \\ x_{10} & x_{14} & x_2 & x_6 \\ x_{15} & x_3 & x_7 & x_{11} \end{bmatrix} \xrightarrow{MC} \begin{bmatrix} x_{16} & x_{20} & x_{24} & x_{28} \\ x_{17} & x_{21} & x_{25} & x_{29} \\ x_{18} & x_{22} & x_{26} & x_{30} \\ x_{19} & x_{23} & x_{27} & x_{31} \end{bmatrix} \xrightarrow{next\ \ round} \\
&\begin{bmatrix} x_{16} & x_{20} & x_{24} & x_{28} \\ x_{17} & x_{21} & x_{25} & x_{29} \\ x_{18} & x_{22} & x_{26} & x_{30} \\ x_{19} & x_{23} & x_{27} & x_{31} \end{bmatrix} \xrightarrow{SC} \begin{bmatrix} x_{16} & x_{20} & x_{24} & x_{28} \\ x_{17} & x_{21} & x_{25} & x_{29} \\ x_{18} & x_{22} & x_{26} & x_{30} \\ x_{19} & x_{23} & x_{27} & x_{31} \end{bmatrix} \xrightarrow{SR} \begin{bmatrix} x_{16} & x_{20} & x_{24} & x_{28} \\ x_{21} & x_{25} & x_{29} & x_{17} \\ x_{26} & x_{30} & x_{18} & x_{22} \\ x_{31} & x_{19} & x_{23} & x_{27} \end{bmatrix} \xrightarrow{MC} \begin{bmatrix} x_{32} & x_{36} & x_{40} & x_{44} \\ x_{33} & x_{37} & x_{41} & x_{45} \\ x_{34} & x_{38} & x_{42} & x_{46} \\ x_{35} & x_{39} & x_{43} & x_{47} \end{bmatrix} \xrightarrow{next\ \ round} \\
&\begin{bmatrix} x_{32} & x_{36} & x_{40} & x_{44} \\ x_{33} & x_{37} & x_{41} & x_{45} \\ x_{34} & x_{38} & x_{42} & x_{46} \\ x_{35} & x_{39} & x_{43} & x_{47} \end{bmatrix} \xrightarrow{SC} \begin{bmatrix} x_{32} & x_{36} & x_{40} & x_{44} \\ x_{33} & x_{37} & x_{41} & x_{45} \\ x_{34} & x_{38} & x_{42} & x_{46} \\ x_{35} & x_{39} & x_{43} & x_{47} \end{bmatrix} \xrightarrow{SR} \begin{bmatrix} x_{32} & x_{36} & x_{40} & x_{44} \\ x_{37} & x_{41} & x_{45} & x_{33} \\ x_{42} & x_{46} & x_{34} & x_{38} \\ x_{47} & x_{35} & x_{39} & x_{43} \end{bmatrix} \xrightarrow{MC} \begin{bmatrix} x_{48} & x_{52} & x_{56} & x_{60} \\ x_{49} & x_{53} & x_{57} & x_{61} \\ x_{50} & x_{54} & x_{58} & x_{62} \\ x_{51} & x_{55} & x_{59} & x_{63} \end{bmatrix} \xrightarrow{next\ \ round} \\
&\begin{bmatrix} x_{48} & x_{52} & x_{56} & x_{60} \\ x_{49} & x_{53} & x_{57} & x_{61} \\ x_{50} & x_{54} & x_{58} & x_{62} \\ x_{51} & x_{55} & x_{59} & x_{63} \end{bmatrix} \xrightarrow{SC} \begin{bmatrix} x_{48} & x_{52} & x_{56} & x_{60} \\ x_{49} & x_{53} & x_{57} & x_{61} \\ x_{50} & x_{54} & x_{58} & x_{62} \\ x_{51} & x_{55} & x_{59} & x_{63} \end{bmatrix} \xrightarrow{SR} \begin{bmatrix} x_{48} & x_{52} & x_{56} & x_{60} \\ x_{53} & x_{57} & x_{61} & x_{49} \\ x_{58} & x_{62} & x_{50} & x_{54} \\ x_{63} & x_{51} & x_{55} & x_{59} \end{bmatrix} \xrightarrow{MC} \begin{bmatrix} x_{64} & x_{68} & x_{72} & x_{76} \\ x_{65} & x_{69} & x_{73} & x_{77} \\ x_{66} & x_{70} & x_{74} & x_{78} \\ x_{67} & x_{71} & x_{75} & x_{79} \end{bmatrix}
\end{aligned} \quad (14)$$

LED 算法 4 轮运算的 MILP 模型主模块 C 语言实现代码描述如下：

```
int main()
{
  int a[4][4];
  for (j = 0; j < 4; j++)
    for (i = 0; i < 4; i++)
      a[j][i] = next++;
    printf("Minimize\n");
    /*输出目标函数*/
    for (j = 0; j < ROUNDS*16-1; j++)
      printf("x%j + ",j);/*ROUNDS 是轮数*/
      printf("x%j\n\n",ROUNDS*16-1);
      printf("Subject To\n");
    /*轮函数约束*/
    for (r = 0; r<ROUNDS; r++)
      { ShiftRows(a);
      MixColumnsSerial(a); }
    /*至少有一个 S 盒是活跃的*/
    for (j = 0; j < ROUNDS*16-1; j++) printf("x%j + ",j);
    printf("x%j >= 1\n\n",
    ROUNDS*16-1);
    printf("Binary\n"); /*  变量约束  */
    for (j = 0; j < 16; j++) printf("x%j\n",j);
      for (j = 0; j < dummy; j++) printf("d%j\n",j);
        printf ("End\n");
  return 0;
}
```

4 MILP 模型求解 LED 算法活跃 S 盒

通过建立 LED 算法半字节的自动化 MILP 模型，并将模型进行 C 语言编程实现，对算法的 4 轮运算进行每一轮自动

化约束，4 轮约束不等式如下：

LED 算法的第 1 轮运算约束不等式：

$$\begin{cases} x_0+x_5+x_{10}+x_{15}+x_{16}+x_{17}+x_{18}+x_{19}-5d_0\ge 0 \\ d_0-x_i\ge 0 \quad (i=0,5,10,15,16,17,18,19) \end{cases} \tag{15}$$

$$\begin{cases} x_1+x_6+x_{11}+x_{12}+x_{20}+x_{21}+x_{22}+x_{23}-5d_1\ge 0 \\ d_1-x_i\ge 0 \quad (i=1,6,11,12,20,21,22,23) \end{cases} \tag{16}$$

$$\begin{cases} x_2+x_7+x_8+x_{13}+x_{24}+x_{25}+x_{26}+x_{27}-5d_2\ge 0 \\ d_2-x_i\ge 0 \quad (i=2,7,8,13,24,25,26,27) \end{cases} \tag{17}$$

$$\begin{cases} x_3+x_4+x_9+x_{14}+x_{28}+x_{29}+x_{30}+x_{31}-5d_3\ge 0 \\ d_3-x_i\ge 0 \quad (i=3,4,9,14,28,29,30,31) \end{cases} \tag{18}$$

LED 算法的第 2 轮算法约束不等式：

$$\begin{cases} x_{16}+x_{21}+x_{26}+x_{31}+x_{32}+x_{33}+x_{34}+x_{35}-5d_4\ge 0 \\ d_4-x_i\ge 0 \quad (i=16,21,26,31,32,33,34,35) \end{cases} \tag{19}$$

$$\begin{cases} x_{20}+x_{25}+x_{30}+x_{19}+x_{36}+x_{37}+x_{38}+x_{39}-5d_5\ge 0 \\ d_5-x_i\ge 0 \quad (i=20,25,30,19,36,37,38,39) \end{cases} \tag{20}$$

$$\begin{cases} x_{24}+x_{29}+x_{18}+x_{23}+x_{40}+x_{41}+x_{42}+x_{43}-5d_6\ge 0 \\ d_6-x_i\ge 0 \quad (i=24,29,18,23,40,41,42,43) \end{cases} \tag{21}$$

$$\begin{cases} x_{28}+x_{17}+x_{22}+x_{27}+x_{44}+x_{45}+x_{46}+x_{47}-5d_7\ge 0 \\ d_7-x_i\ge 0 \quad (i=28,17,22,27,44,45,46,47) \end{cases} \tag{22}$$

LED 算法的第 3 轮运算约束不等式：

$$\begin{cases} x_{32}+x_{37}+x_{42}+x_{47}+x_{48}+x_{49}+x_{50}+x_{51}-5d_8\ge 0 \\ d_8-x_i\ge 0 \quad (i=32,37,42,47,48,49,50,51) \end{cases} \tag{23}$$

$$\begin{cases} x_{36}+x_{41}+x_{46}+x_{35}+x_{52}+x_{53}+x_{54}+x_{55}-5d_9\ge 0 \\ d_9-x_i\ge 0 \quad (i=36,41,46,35,52,53,54,55) \end{cases} \tag{24}$$

$$\begin{cases} x_{40}+x_{45}+x_{34}+x_{39}+x_{56}+x_{57}+x_{58}+x_{59}-5d_{10}\ge 0 \\ d_{10}-x_i\ge 0 \quad (i=40,45,34,39,56,57,58,59) \end{cases} \tag{25}$$

$$\begin{cases} x_{44}+x_{33}+x_{38}+x_{43}+x_{60}+x_{61}+x_{62}+x_{63}-5d_{11}\ge 0 \\ d_{11}-x_i\ge 0 \quad (i=44,33,38,43,60,61,62,63) \end{cases} \tag{26}$$

LED 算法的第 4 轮运算约束不等式：

$$\begin{cases} x_{48}+x_{53}+x_{58}+x_{63}+x_{64}+x_{65}+x_{66}+x_{67}-5d_{12}\ge 0 \\ d_{12}-x_i\ge 0 \quad (i=48,53,58,63,64,65,66,67) \end{cases} \tag{27}$$

$$\begin{cases} x_{52}+x_{57}+x_{62}+x_{51}+x_{68}+x_{69}+x_{70}+x_{71}-5d_{13}\ge 0 \\ d_{13}-x_i\ge 0 \quad (i=52,57,62,51,68,69,70,71) \end{cases} \tag{28}$$

$$\begin{cases} x_{56}+x_{61}+x_{50}+x_{55}+x_{72}+x_{73}+x_{74}+x_{75}-5d_{14}\ge 0 \\ d_{14}-x_i\ge 0 \quad (i=56,61,50,55,72,73,74,75) \end{cases} \tag{29}$$

$$\begin{cases} x_{60}+x_{49}+x_{54}+x_{59}+x_{76}+x_{77}+x_{78}+x_{79}-5d_{15}\ge 0 \\ d_{15}-x_i\ge 0 \quad (i=60,49,54,59,76,77,78,79) \end{cases} \tag{30}$$

根据 LED 算法的 4 轮运算约束不等式，进行求解算法前 4 轮差分活跃 S 盒，在 MILP 自动化分析模型中，r 轮差分特征至少包含一个 S 盒是活跃 S 盒，根据算法的第 1 轮约束不等式，设一个半字节的差分变量 $x_0\ne 0$，经过算法的 S 盒替换变换，由于 LED 算法的 S 盒是一个双射的 S 盒，当输入差分是非零的，则输出差分也是非零，从而 LED 算法第 1 轮运算得到 1 个差分活跃 4×4 S 盒，再经过行移位变换与列混合变换，半字节的差分变量由 $x_0,x_1,\ldots,x_{14},x_{15}$ 变为 $x_{16},x_{17},\ldots,x_{30},x_{31}$，根据约束不等式（15），当半字节的差分变量 $x_0\ne 0$，则变量 $d_0\ne 0$；为了满足约束不等式（15）成立，则

$$\begin{cases} x_{16}\ne 0 \\ x_{17}\ne 0 \\ x_{18}\ne 0 \\ x_{19}\ne 0 \end{cases} \tag{31}$$

同时，其他半字节差分变量为零，使得约束不等式（16）～（18）成立。

在第 2 轮约束不等式中，由于第 1 轮运算中 4 个差分变量 $x_{16},x_{17},x_{18},x_{19}$ 都不为 0，经过算法的 S 盒替换变换，则 LED 算法第 2 轮运算得到 4 个差分活跃 S 盒，再经过行移位变换与列混合变换，半字节的差分变量由 $x_{16},x_{17},\ldots,x_{30},x_{31}$ 变为 $x_{32},x_{33},\ldots,x_{46},x_{47}$，根据约束不等式（19）～（22），4 个差分变量 $x_{16},x_{17},x_{18},x_{19}$ 都不为 0，则变量 d_4,d_5,d_6,d_7 都不为 0；为了满足约束不等式（19）～（22）成立，则

$$\begin{cases} x_{32}\ne 0 \\ x_{33}\ne 0 \\ x_{34}\ne 0 \\ x_{35}\ne 0 \end{cases}, \begin{cases} x_{36}\ne 0 \\ x_{37}\ne 0 \\ x_{38}\ne 0 \\ x_{39}\ne 0 \end{cases}, \begin{cases} x_{40}\ne 0 \\ x_{41}\ne 0 \\ x_{42}\ne 0 \\ x_{43}\ne 0 \end{cases}, \begin{cases} x_{44}\ne 0 \\ x_{45}\ne 0 \\ x_{46}\ne 0 \\ x_{47}\ne 0 \end{cases} \tag{32}$$

在第 3 轮约束不等式中，由于第 2 轮运算中 16 个差分变量 $x_{32},x_{33},\ldots,x_{46},x_{47}$ 都不为 0，经过算法的 S 盒替换变换，则 LED 算法第 3 轮运算得到 16 个差分活跃 S 盒，再经过行移位变换与列混合变换，半字节的差分变量由 $x_{32},x_{33},\ldots,x_{46},x_{47}$ 变为 $x_{48},x_{49},\ldots,x_{62},x_{63}$，根据约束不等式（23）～（26），16 个差分变量 $x_{32},x_{33},\ldots,x_{46},x_{47}$ 都不为 0，则变量 d_8,d_9,d_{10},d_{11} 都不为 0；为了满足不等式（23）～（26）成立，则选择

$$\begin{cases} x_{48}\ne 0 \\ x_{53}\ne 0 \\ x_{58}\ne 0 \\ x_{63}\ne 0 \end{cases} \tag{33}$$

在第 4 轮约束不等式中，由于第 3 轮运算中 4 个差分变量 $x_{48},x_{53},x_{58},x_{63}$ 都不为 0，经过算法 S 盒替换变换，则 LED 算法第 4 轮运算得到 4 个差分活跃 S 盒，再经过行移位变换与列混合变换，半字节的差分变量由 $x_{48},x_{49},\ldots,x_{62},x_{63}$ 变为 $x_{64},x_{65},\ldots,x_{78},x_{79}$，根据约束不等式（23）～（26），4 个差分变量 $x_{48},x_{53},x_{58},x_{63}$ 都不为 0，则变量 d_{12} 不为 0；为了满足约束不等式（27）成立，则可以选择半字节差分变量 x_{64} 不为 0，同时，其他半字节差分变量为零，使得约束不等式（28）～（30）成立。

经过第 4 轮约束不等式的分析之后，则在第 4 轮中又恢复到只有一个半字节差分变量不为 0，其他半字节差分变量为 0；从而算法第 5 轮的变量与约束不等式与第 1 轮的情况相同，从而 LED 算法后续运算变换出现了像前 4 轮一样的周期性变化，所以，本文只需通过计算前 4 轮的活跃 S 盒，寻找周期变化规律，可以精确地推算出全轮 LED 算法的活跃 S 盒数量。LED 算法前 4 轮计算得到差分活跃 S 盒至少为 1+4+16+4=25 个。

建立 LED 算法的自动化 MILP 模型进行差分分析，分析过程中，半字节差分变量个数、约束不等式数量及活跃 S 盒数量随着算法加密轮数递增的变化情况如表 2 所示，列举了算法 14 轮加密。

表 2 MILP 模型的 LED 算法差分分析

Table 2 MILP module for differential cryptanalysis of LED

轮数	变量数	不等式数	活跃 S 盒数
1	32	36	1
2	48	72	5
3	56	108	21
4	64	144	25
5	80	180	26
6	96	216	30
7	112	252	46
8	128	288	50
9	140	324	51
10	156	360	55
11	172	396	71
12	188	432	75
13	204	468	76
14	220	504	80

从表 2 可以看出，r 轮 LED 算法的 MILP 模型差分分析规模为 $16+16\times r$ 个半字节差分变量和 $32\times r$ 个约束不等式，从而 LED-64 算法活跃 S 盒数量变化情况为表 3 所示，LED-128 算法活跃 S 盒数量变化情况为表 4 所示。

表 3 LED-64 算法活跃 S 盒数量

Table 3 The number of active S-boxes in LED-64

轮数	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
数量	1	5	21	25	26	30	46	50	51	55	71	75	76	80	96	100
轮数	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
数量	101	105	121	125	126	130	146	150	151	155	171	175	176	180	196	200

表 4 LED-128 算法活跃 S 盒数量

Table 4 The number of active S-boxes in LED-128

轮数	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
数量	1	5	21	25	26	30	46	50	51	55	71	75	76	80	96	100
轮数	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
数量	101	105	121	125	126	130	146	150	151	155	171	175	176	180	196	200
轮数	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
数量	201	205	221	225	226	230	246	250	251	255	271	275	276	280	296	300

通过分析表 3 和 4 的结果，算法 4 轮加密运算是包含至少 25 个活跃 S 盒，LED-64 算法全轮是至少有 200 个活跃 S 盒，LED-128 算法全轮是至少有 300 个活跃 S 盒。这个结果与 LED 算法设计者给出的活跃 S 盒理论值是相同，从而验证了本文提出的基于半字节 MILP 自动化搜索模型是正确的。LED 算法的 S 盒的差分概率是 2^{-2}，线性概率也是 2^{-2}，从而，计算 LED 算法的差分特征与线性逼近，LED-64 算法的最大差分特征概率计算为 $2^{-2\times200}=2^{-400}$，根据堆积定理[10]最大线性逼近概率计算为 $2^{200-1}\times2^{-2\times200}=2^{-201}$，LED-128 算法的最大差分特征概率计算为 $2^{-2\times300}=2^{-600}$，最大线性逼近概率计算为 $2^{300-1}\times2^{-2\times300}=2^{-301}$。通过计算与分析，证明了 LED 算法是很好抵抗差分与线性攻击。

5 结束语

本文提出一种面向半个字节 MILP 模型自动化搜索密码算法的差分特征与线性逼近，将该方法结合 LED 算法的算法结构，进行 C 语言编程实现。在该模型中，以较少的变量与约束不等式求解算法的活跃 S 盒数量，精确地求解 LED 算法 4 轮运算至少包含 25 个活跃 S 盒，LED-64 算法全轮是至少有 200 个活跃 S 盒，LED-128 算法全轮是至少有 300 个活跃 S 盒。这个结果与 LED 算法设计者给出的活跃 S 盒理论值是相同，从而验证了本文提出的面向半字节 MILP 自动化搜索模型是正确的。LED-64 算法的最大差分特征概率为 2^{-400}，最大线性逼近概率为 2^{-201}，LED-128 算法的最大差分特征概率为 2^{-600}，最大线性逼近概率为 2^{-301}。通过计算与分析，证明了 LED 算法是可以抵抗差分与线性攻击。

在下一步工作中，将提出的一种面向半个字节 MILP 模型自动化搜索密码算法的差分特征与线性逼近方法，应用到更多分组密码算法中，总结出通用高效的自动化 MILP 模型。

参考文献：

[1] Bogdanov A, Knudsen L R, Leander G, *et al*. PRESENT: an ultra-lightweight block cipher [C]//Proc of International Workshop on Cryptographic Hardware and Embedded Systems. Berlin:Springer, 2007: 450-466.

[2] Guo Jian, Peyrin T, Poschmann A, *et al*. The LED block cipher [C]//Proc of International Conference on Cryptographic Hardware and Embedded Systems. Berlin:Springer-Verlag, 2011: 326-341.

[3] Wu Wenling, Zhang Lei. LBlock: a lightweight block cipher [C]// Applied Cryptography and Network Security. Berlin:Springer, 2011: 327-344.

[4] Gong Zheng, Nikova S, Law Y W. KLEIN: a new family of lightweight block ciphers [C]//Proc of International Conference on RFID Security and Privacy. Springer-Verlag, 2011: 1-18.

[5] Zhang Wentao, Bao Zhenzhen, Lin Dongdai, *et al*. RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms [J]. Science China, 2015, 58(12): 122103-122103.

[6] Beierle C, Jean J, Kölbl S, *et al*. The SKINNY, Family of Block Ciphers and Its Low-Latency Variant MANTIS [C]//Advances in Cryptology. Berlin:Springer, 2016: 123-153.

[7] Li Lang, Liu Botao, Zhou Yimeng, *et al*. SFN: a new lightweight block cipher [J]. Microprocessors & Microsystems, 2018, 60: 138-150.

[8] Wang Ding, Li Wenting, Wang Ping. Measuring two-factor authentication schemes for real-time data access in industrial wireless sensor networks [J]. IEEE Trans on IndustrialInformatics, 2018, 99: 1-12.

[9] Li Wei, Zhang Wenwen, Gu Dawu, *et al*. Impossible differential fault analysis on the LED lightweight cryptosystem in the vehicular Ad-hoc networks [J]. IEEE Trans on Dependable & Secure Computing, 2016, 13 (1): 84-92.

[10] Wang Ding, Cheng Haibo, He Debiao, *et al*. On the challenges in designing identity-based privacy-preserving authentication schemes for mobile devices [J]. IEEE Systems Journal, 2018, 21(1): 916-925.

[11] Biham E, Shamir A. Differential cryptanalysis of DES-like cryptosystems [J]. Journal of Cryptology, 1991, 4(1): 3-72.

[12] Matsui M. Linear Cryptanalysis Method for DES Cipher [C]//Proc of Eurocrypt-Advances in Cryptology. 1993, 765: 386-397.

[13] Mouha N, Wang Qingju, Gu Dawu, *et al*. Differential and linear cryptanalysis using mixed-integer linear programming [C]// Proc of International Conference on Information Security and Cryptology. Springer-Verlag, 2011: 57-76.

[14] Sun Siwei, Hu Lei, Wang Peng, *et al*. Automatic security evaluation and (related-key) differential characteristic search: application to SIMON, PRESENT, LBlock, DES (L) and other bit-oriented block ciphers [C]// Advances in Cryptology–ASIACRYPT. Berlin:Springer, 2014: 158-178.

[15] Fu Kai, Wang Meiqin, Guo Yinghua, *et al*. MILP-based automatic search algorithms for differential and linear trails for speck [C]// Fast Software Encryption. Berlin:Springer, 2016: 268-288.

[16] Beaulieu R, Treatman-Clark S, Shors D, *et al*. The SIMON and SPECK lightweight block ciphers [C]//Proc of Design Automation Conference. IEEE, 2015: 1-6.

[17] 尹军，马楚焱，宋健，等. 轻量级分组密码算法 ESF 的安全性分析 [J]. 计算机研究与发展，2017, 54(10): 2224-2231. (Yin Jun, Ma Chuyan, Song Jian, *et al*. Security Analysis of lightweight block cipher ESF [J]. Journal of Computer Research and Development, 2017, 54(10): 2224-2231.)

[18] 尹军，宋健，曾光，等. 轻量级分组密码算法 ESF 的相关密钥差分分析 [J].密码学报,2017,4(4): 333-344. (Yin Jun, Song Jian, Zeng Guang, *et al*. Related-key differential attack on lightweight block cipher ESF [J]. Journal of Cryptologic Research, 2017, 4 (4): 333-344.)

[19] Liu Xuan, Zhang Wenying, Liu Xiangzhong, *et al*. Eight-sided fortress: a lightweight block cipher [J]. Journal of China Universities of Posts and Telecommunications, 2014, 21(1): 104-108.

[20] 李玮，谷大武，赵辰，等. 物联网环境下 LED 轻量级密码算法的安

全性分析 [J]. 计算机学报, 2012, 35(3): 434-445. (Li Wei, Gu Dawu, Zhao Chen, *et al.* Security analysis of the LED lightweight cipher in the Internet of things [J]. Chinese Journal of Computers, 2012, 35(3): 434-445.)